Amendments to the Claims:

This listing of claims replaces all prior versions and listings of claims in the application:

Listing of Claims:

1. (Currently Amended) A system comprising:

a parallel processor that assigns system functions for processing data, the parallel processor comprising:

a plurality of programming engines that support multiple contexts, the plurality of programming engines arranged to provide a functional pipeline; [[and]]

a functional pipeline control unit that assigns system ~~function~~ functions to and passes functional data among the plurality of programming engines; and

a critical section that provides exclusive privileges to one of the multiple contexts for modifying a shared resource.

2. (Currently Amended) The system of claim 1 further comprising:

a synchronization unit across the functional pipeline unit for synchronizing tasks performed by the programming engines.

3. (Original) The system of claim 1 wherein the functional pipeline unit includes a plurality of functional pipeline stages.

4. (Original) The system of claim 3 wherein the plurality of programming engines have an execution time for processing a task and the execution time is partitioned into a number of time intervals corresponding to the number of the plurality of functional pipeline stages.

5. (Previously Presented) The system of claim 4 wherein each of the plurality of functional pipeline stages performs a different system function.

6. (Original) The system of claim 1 wherein at least one of the plurality of programming engines is the functional pipeline unit.

7. (Original) The system of claim 1 wherein the plurality of programming engines are configured to process a data packet in order.

8. (Original) The system of claim 7 wherein the data packet are assigned to the multiple contexts of the plurality of programming engines.

9. (Original) The system of claim 1 wherein the plurality of programming engines are configured to execute a data packet processing function using the functional pipeline unit of the system.

10. (Original) The system of claim 9 wherein a data packet is maintained in the plurality of programming engines for a period of time corresponding to the number of the plurality of programming engines.

11. (Original) The system of claim 3 wherein the number of the plurality of pipeline stages is equal to the number of the plurality of programming engines.

12. (Currently Amended) The system of claim 3 wherein the plurality of pipeline stages include [[a]] the critical section.

13. (Original) The system of claim 12 wherein the critical section provides exclusive access for the multiple contexts to non-shared data required for processing data packets.

14. (Original) The system of claim 3 wherein the plurality of programming engines include inter-thread signaling.

15. (Original) The system of claim 3 wherein the plurality of programming engines include an elasticity buffer that accommodates jitter between the plurality of pipeline stages upon execution of a data packet processing function.

16. (Currently Amended) A method of transferring data between a plurality of programming engines, the method comprising:

assigning system functions for processing data to corresponding ones of a plurality of programming engines that provide a functional pipeline unit in a parallel processor and, which supports execution of multiple contexts in each of the plurality of programming engines; [[and ]]

passing functional data among the plurality of programming engines in the functional pipeline unit; and providing exclusive privileges to one of the multiple contexts for modifying a shared resource.

17. (Currently Amended) The method of claim 16 further comprising:

synchronizing the system functions ~~accross~~ from a first programming engine of the functional pipeline unit to a second programming engine of the functional pipeline unit.

18. (Previously Presented) The method of claim 17 further comprising:

partitioning an execution time into a number of time intervals corresponding to the number of plurality of pipeline stages.

19. (Original) The method of claim 16 wherein the plurality of programming engines use multiple contexts to process the data packet in order.

20. (Original) The method of claim 16 wherein the plurality of programming engines execute a data packet processing functions using the functional pipeline unit of the system.

21. (Previously Presented) The method of claim 16 further comprising:

using a critical section that provides exclusive access for the multiple contexts to non-shared data required for processing data packets.

22. (Previously Presented) The method of claim 16 further comprising:

employing an elasticity buffer to accommodate jitter between the plurality of pipeline stages upon execution of a data packet processing function.

23. (Currently Amended) A computer program product residing on a computer readable medium for causing a parallel processor to perform a function comprises instructions causing the processor to:

assign system functions for processing data in a parallel processor to corresponding ones of a plurality of programming engines that provide a functional pipeline unit, which supports execution of multiple contexts in each of the plurality of programming engines; [[and ]]

pass functional data among the plurality of programming engines in the functional pipeline unit; and

provide exclusive privileges to one of the multiple contexts for modifying a shared resource.

24. (Currently Amended) The computer program product of claim 23 further comprising instructions causing the processor to synchronize the system functions ~~accross~~ from a first programming engine of the functional pipeline unit to a second programming engine of the functional pipeline unit.

25. (Original) The computer program product of claim 23 wherein the plurality of programming engines execute a data packet processing functions using the functional pipeline unit of the system.